APPLICATION UNDER UNITED STATES PATENT LAWS

Atty. Dkt. No.:	2540-0664
Invention:	**VIDEO COMPRESSION SYSTEM**
Inventor(s):	Timothy A. Johnson

Davidson Berquist
Klima & Jackson LLP
4501 North Fairfax Drive
Suite 920
Arlington, VA 22203
(703) 248-0333 Phone
(703) 248-9558 Fax

This is a:

☐ Provisional Application

☒ Regular Utility Application

☐ Continuing Application
____ The contents of the parent are incorporated by reference

☐ PCT National Phase Application

☐ Design Application

☐ Reissue Application

☐ Plant Application

☐ PCT International Phase Application

SPECIFICATION

TITLE OF THE INVENTION

VIDEO COMPRESSION SYSTEM

FIELD OF THE INVENTION

[0001] This invention relates to video signal compression techniques.

BACKGROUND OF THE INVENTION

[0002] In its raw form, video is a high volume form of data. Communication bandwidth in its variety of forms can be quickly consumed by the transmission of raw video information. Video compression techniques are known that attempt to retain the highest reproducibility of the original video while reducing the data volume required to transmit the information.

[0003] One form of video compression involves run length encoding in which patterns of pixels in a serial run of pixel information are identified, encoded, and transmitted by the code rather than by the individual pixels. U.S. Patent Application No. 10/260,534, entitled "Video Compression System," filed on October 1, 2002 (the '534 Application) describes in some detail various kinds of run length encoding combined with some other types of novel pattern encoding. The entire contents of the '534 application are incorporated herein by reference. The algorithms referred to in the '534 application are sometimes referred to herein as "DVC compression."

[0004] Another form of video compression divides video information into larger blocks of pixels and then transmits information about the block of pixels, rather than about the individual pixels themselves. In such systems, video received by a computer or other video source is usually loaded into a frame buffer. From there, blocks of information are compared to corresponding prior frame blocks to determine whether changes have occurred within the blocks. Video compression occurs because information about blocks without detected change need not be transmitted to the receiving end. Instead, the receiving end, employing the known compression algorithm used at the compression end, assumes that the pixels within a block remain the same from frame-to-frame until told otherwise by the compression end.

[0005] In an example block-by-block algorithm 8-bit video is captured and divided into 64x16 bit blocks of video for comparison with corresponding prior blocks. When video in a block is detected as changed, the information for that block is communicated to the receiving end. If no change is detected, no transmission occurs with respect to that block.

[0006] Such algorithms are presently employed in the computer environment to permit remote access to computer systems from non-proximate workstations. Keyboard, mouse and other (low volume) serial device information is communicated from the workstation to the remote computer via a communication link, and resultant (potentially high volume) video information is communicated back from the computer to the workstation via the communication link. Such systems and the methods they employ for such communications are by now well-known to the artisan.

[0007] In that computer environment, it is important that the computer video not "lag" behind the user's keyboard and mouse movements. It is unacceptable for a user to hit a key or move a mouse and see the results of that on the workstation monitor with any perceptible lag. In the remote computer environment, that means that the serial device information must communicated to the remote computer (usually a relatively easy task due to the low data volume), the computer's video processor must act on the information to create new video, and the computer's communication system must communicate the video back to the user—all before the user can detect lag between the moment the keyboard/mouse is used and the moment the resultant video is returned.

[0008] The block-by-block video encoding is one way that the video from the computer to the workstation is moved quickly back to the workstation such that no perceptible lag exists to the user. In the computer environment, many blocks within the computer screen of information may not change from frame to frame (such as on a desktop background, word processing document background, or other "inactive" location of the computer screen). With the block-by-block encoding, such blocks could be effectively ignored by the computer when sending the video information to the workstation because the workstation would assume that the blocks were not changed unless specifically told that they had been.

[0009] Block analysis occurred by evaluating pixel values within the block. One method provided a transmission of the block whenever a block change was detected, but a more improved method provided a number of pixels that had to change before the block transmission occurred. Thus, in the situation where blocks were 64x16, for example, a threshold number of six of the 1024 pixels had to change before the block was considered different enough to warrant the block transmission. The threshold gave the system room to accommodate a noise floor such that blocks could be ignored unless a change in the number of pixels exceeded a number that might be expected in a typical, noisy environment.

[0010] The block analysis was very effective in eliminating lag between keyboard/mouse entry and the corresponding video presentation. A problem developed, however, in which blocks would experience an important substantive pixel change that would go unreported to the workstation because the substantive pixel change did not cause the block to the exceed the noise threshold of, for example, six pixels changed. Examples of such occurrences included the tip of a mouse pointer moving through one or two pixels of a block. Because the number of pixels in the block that changed (the one or two pixels defining a piece of the mouse tip, or perhaps even three or four pixels once some noisiness was taken into consideration) failed to exceed the threshold, the block changes were unreported to the workstation. The effect was a lack of clarity in the mouse tip displayed at the workstation. The same situation occurred with respect to characters in a word processing document or on the desktop. If a two or three pixels of a changed character, such as the end of the hook in an "r" stretched into a block that was otherwise unchanged, the pixels would not be communicated and the end of the "r" would be deleted.

[0011] So, if the number of pixels changed within the block was set to one for the changes to be communicated, noise within the system would cause truly unchanged blocks to be communicated anyway. But, if the number of pixels changed within the block was set to more than one, potentially important pixel changes were lost.

BRIEF SUMMARY OF THE INVENTION

[0012] Pixels are defined as N-bits of code defining the red hue, N-bits defining the green hue, and N-bits defining the blue hue of the pixel. In the prior system, if the block analysis detected any change in any of the N-bits of any of the three hues of any of the pixels in the block (for example, 1000 pixels in the 64x16 block), then the pixel was counted as changed for purposes of meeting the threshold number of pixels that had to change before the block was communicated. In the present embodiment, two different analyses are conducted on each block. First, a traditional delta is applied to the number of pixels in a block that change. The delta is referred to herein as a block change threshold and describes the number of pixels within a defined block that change from one reference (such as a frame) to another point (such as a subsequent frame). Thus, if any of the 3N bits in the RGB code change, the pixel is considered changed and counted for purposes of calculating the block change. If the block change exceeds a threshold, the block change is communicated to the receiving end.

[0013] The second analysis evaluates the contrast change of each independent pixel within the block. This second delta is referred to herein as a pixel change threshold and describes the extent in value to which a pixel change occurs. High contrast pixel changes will cause a block to be communicated to the receiving end—even though the block change threshold is not met.

[0014] The two-part analysis solves both of the problems characteristic of prior block analyses. First, if change occurs in a block solely because of pixel noise, the pixel change should be of low value and will register as a single pixel change for purposes of the block change calculation. If it is the only pixel change, the block change threshold will not be met, nor will the pixel change threshold (because the change in value is low contrast, i.e., low value). Accordingly, the noise will not cause the block to be transmitted, thereby conserving bandwidth. But, if the only pixel change is a high contrast change, then the block change threshold will not be met but the pixel change threshold will. Accordingly, the block will be transmitted even though the block change threshold was not met. Similarly, if a sweeping subtle color change occurs in the corner of a block, the pixel change threshold will not be met but the block change threshold (for number of changed pixels) will. Accordingly, the block will be appropriately communicated to reflect the subtle color change.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] FIGURE 1 is a schematic representation of an example system into which the present invention can have application;

[0016] FIGURE is an example video frame; and

[0017] FIGURE 3 is an example embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0018] In the computer system 10 of Figure 1, keyboard and mouse information 17 is entered at the client 12, processed by the processor 19 at the client 12, and communicated to the server 11 via the communication channel 16. The communication channel can be a telephone line, a LAN, a WAN, the Internet, a fiber link, a wireless link or any other suitable communication system. The server 11 receives the keyboard and mouse information 17, processes it in processor 13 in accordance with an application of the server 11 and prepares an appropriate video response. The video information 18 is communicated to the client 12, which receives it, processes it in processor 19 and displays it for the client user.

[0019] The typical client user will not tolerate much delay between the time the user strikes a key or moves the mouse and the time that that user sees the resultant video on the monitor (not shown) of the client 12. Thus, the loop of information between the client 12 and server 11, together with the above-described processing, must not introduce perceptible lag. Fortunately, hardware and software for processors 19 and 13 are available to rapidly process the keyboard, mouse and video information, without introducing delay perceptible to the user. The communication channel 16, on the other hand, may impose perceptible delay unless the video information 18 is satisfactorily compressed.

[0020] For that reason, video encoder 14 and video decoder 15 are provided at, respectively, the server 11 and client 12. The video encoder 14 and decoder 15 can be a system such as that described in the '534 application referred to in the background above, with modifications in accordance with the following descriptions.

[0021] As described above, the encoder 14 can be a block encoder in which blocks of video information are analyzed for change to determine whether communication of the block is necessary for a particular frame. In accordance with the present invention, that determination occurs on two fronts, one based on a pixel change threshold and the other based on a block change threshold.

[0022] In the block change threshold determination, the number of pixels that change in value within the block are counted and compared to a threshold. If the number exceeds the threshold, the block is considered sufficiently changed to warrant transmission.

[0023] In the pixel change threshold determination, the value of change in each pixel is analyzed and the change values compared to a threshold. If the change value of any pixel in the block exceeds the threshold, the block is considered sufficiently changed to warrant transmission.

[0024] Transmission of "the block" can take a variety of forms, all of which are encompassed within the present invention. In the basic form, the raw pixel values for the block are communicated in serial fashion. In a more preferable form, the serial pixel values for the block are passed through the DVC compression technique of the '534 application for further compression. Other forms of compression are equally plausible. The present invention is not limited to any particular manner in which the block (once identified as a candidate for transmission) is transmitted from the server 11 via the channel 16 to the client 12.

[0025] Figure 2 illustrates an example of why the present invention can be so valuable. In Figure 2, a video screen 20 is shown in small part, with a mouse cursor 21 illustrated on the screen 20. Internally to the present system, the screen of information is divided into blocks, such as blocks 22-25. Assuming for the purpose of illustration only, that the cursor 21 is the only thing to be changed on the client user's screen and that it is moving straight upward (in the direction of the arrow shown). Assume also that in the frame shown in Figure 2, the portion 21A of the cursor 21 just entered block 25 for the first time. That portion 21A is shown in greater detail in the bottom of Figure 2, where the point of the cursor is drawn by pixels 29, 30, 32 and 33. Surrounding pixels 26, 27, 28, 31, 34 and 35 can be assumed to remain unchanged. As described, the block 25 would *ideally* have four and only four pixel changes, namely pixels 29, 30, 32 and 33 would change from background color to cursor color.

[0026] If the system used the prior art block analysis, the block 25 would not qualify for transmission because the number of changed pixels (four) would not meet the threshold (six) required for the server 11 to communicate the block change. The result means that the user of the client 12 will receive video on the monitor with block 25 unchanged, i.e., the potion 21A of the cursor 21 will be cut off. Such rounded pointer tips and other flaws in the clarity of characters are characteristic of block analysis systems.

[0027] The present invention solves that problem by combining the block threshold analysis with a substantially simultaneous pixel contrast analysis. In accordance with the preferred embodiment, the block 25 of Figure 2 is compared to determine whether enough pixels changed, and if not, each pixel in the block 25 is evaluated to determine whether any one pixel changed *enough* to merit transmission of the block. In Figure 2, the pixels 29, 30, 32, and 33, would all qualify under pixel contrast limits (assuming that the background color is not indistinguishably close to the cursor color).

[0028] The present invention is not delimited by actual number for the block count threshold. As described above, a threshold of six pixels per 64x16 block can be useful. But, so too can thresholds of less than a 6/1000 ratio, even to a threshold of 2 pixels per block of many tens or hundreds of thousands of pixels per block. The actual number is not delimiting, since the artisan with the understanding gained from this disclosure will recognize that more or less sensitivity can be achieved as desired by increasing or decreasing the threshold. Thus, the threshold that provides high noise suppression, such as requiring all of the pixels changed per block before the block is transmitted can be envisioned if, for example, the block is defined as a relatively small 2x2. The present invention is intended to cover all such threshold values, from two pixels to 100% of the pixels in a block.

[0029] Similarly, the actual pixel change threshold is not a limiting factor of the present invention in its broadest forms. Since the transmission decision is a function of both the block change threshold and the pixel change threshold, the two values should preferably be chosen to compliment each other. A relatively lower pixel change threshold causes lower contrast changes in a single pixel to trigger block transmission. A relatively higher pixel change threshold will reduce the possibility of block transmission. Design choices can be made based on such criteria as the noise floor, nominal noise spikes, the number of bits per color component, and the like such that substantive pixel changes are communicated while noise changes are not. The present invention is intended to cover all such threshold values, from a low pixel contrast delta of value = 1 to a high pixel contrast delta of 50% or more of the pixel color value.

[0030] Figure 3 illustrates an example system and methodology for implementing the two-part threshold. Input video 41 is communicated (in 15 bit video in the example illustration of Figure 3) to FIFO 42. FIFO 42 releases the input video to the reference memory 57 where it is stored as frames of video information. Reference memory 57 may be in this example a 1Mx16 memory. The reference memory 57 stores the video information until released by the memory controller 58 via the address and control lines shown. One location that the memory controller 58 releases a frame of information to is the Pixel Difference calculator 43. In the example embodiment where one frame of information is compared to its immediately preceding frame, the memory controller 58 releases the prior frame of information from the memory 57 to the Pixel Difference calculator 43 as the FIFO 42 also communicates the current 15 bit video frame to the Pixel Difference calculator. As corresponding pixels arrive, the Pixel Difference calculator compares the absolute value of a current pixel with its counterpart in the immediately preceding frame. The value of that difference is communicated by the Pixel Difference calculator 43 to the threshold blocks 44 and 46.

[0031] Pixel Threshold Block 44 receives a noise threshold 45 which is compared to each pixel difference value received from the Pixel Difference Block 43. That is, as each pixel difference value (indicating the extent of contrast between the three 5-bit color component values of the current pixel compared to its counterpart in the immediately preceding frame) is received from the Pixel Difference Block 43, it is compared to the noise threshold to determined whether a change in the pixel value is going to be deemed to have occurred. Delta values from the Pixel Difference block 43 that fall below the noise threshold 45 will be presumed to have been unchanged, i.e., system induced rather than substantive screen image changes. The noise threshold can be a five bit signal to evaluate each of the three five bit color component values received as the difference signal from the Pixel Difference Block 43. Block 44 responds with a positive output signal to the Threshold Block Counters 50 each time the difference value received from the Pixel Difference block 43 exceeds the noise threshold. The output signal cnt++ is a count signal received by the Threshold Block Counters 50 which count the number of such signals received.

[0032] Meanwhile, the same delta value for the current pixel is received from the Pixel Difference Block 43 by the Pixel Threshold Block 46. The Pixel Threshold Block 46 compares the delta to another threshold, the "Override Threshold" 47, which it receives as the threshold over which the pixel will be said to have changed so substantively that it must be communicated to the client 12 regardless of other pixel changes within the block. When the Pixel Threshold Block 46 receives a pixel delta value that exceeds that override threshold, the block 46 communicates a number of signals to the Threshold Block Counters 50 equal to the "Count Threshold Number" 52 at which the system will send the block to the client. In other words, in the example of Figure 3, the Threshold Block Counter 50 is tricked into counting the existence of more excessive pixel deltas than really occurred (the "true" number will still be reported by the block 44 to counter 50 although the outcome—block transmission—will have already been predestined by the block 46 report to the counter 50).

[0033] The Counter 50 counts all of the pixel reports from both the Block 44 and the Block 46 for a full block. When the block of pixels begins at the Pixel Difference Block 43, the processor (not shown in Figure 3) enables the counter via enable line 48. At the conclusion of a block, the processor resets the counter to zero by the reset line 49. Thus, the Counter 50 is always counting the excess deltas (or the pseudo-excesses reported by the Block 46 as the case may be) in one block of video information.

[0034] The number of excess deltas counted by the Counter 50 is reported to the Comparators 51, which determine whether the counted number of excess deltas exceeded the count threshold 52 for the block reported. If they did, the Comparators 51 report the block condition (changed or unchanged) to the Block Update Memory 55, which sets a flag for each block that requires updating, i.e., communication to the receiving end. When the frame of input video 41 is completed, the Block Update Memory communicates the block identities that require communication (based on the set flags) to the memory controller 58 which causes a download of those blocks from the Reference Memory to the DVC Compression Block 59.

[0035] Once a block is identified for transmission, the memory 57 provides the blocks that require communication to the DVC Compression Block 59. The DVC Compression Block 59 receives the requisite blocks, compresses the blocks of information (per the '534 Application disclosure), and releases the compressed stream to Output FIFO 60. The compressed stream of video can then go on the Processor Bus 61 for transmission via, for example, a communication channel interface, to the communication channel 16.

[0036] The structure of Figure 3 is not the only embodiment into which the broad aspects of the invention can be incorporated. The structures of Figure 3 can be completely incorporated into software modules, hardware, or a combination of both. The calculations, counting, comparisons and other functions can be performed by discrete units as shown or can be performed as a processing method within a single or multiple processing elements. The present invention is not limited to the particular kind of structure chosen to incorporate the two-threshold analysis.

[0037] Nor is the method of tricking the Counter 50 into a maximum count the only method envisioned by the present invention for determining when a block will be transmitted prior to the true count of pixel changes reaching the count threshold. Other methods could envision a direct reporting of that condition to the memory controller, or a completely independent path of processing elements for analyzing the block count versus the pixel value count.

[0038] While the invention has been described in connection with what is presently considered to be the most practical and preferred embodiment, it is to be understood that the invention is not to be limited to the disclosed embodiment, but on the contrary, is intended to cover various modifications and equivalent arrangements included within the spirit and scope of the appended claims.